Per la webapp su Python Plotly Dash e Docker, è necessario seguire i seguenti passi:

1. Installare e configurare Docker sul proprio sistema.
2. Creare un nuovo progetto Dash utilizzando il codice di esempio fornito dalla documentazione di Dash.
3. Utilizzare un componente di input, come un selettore a discesa, per consentire all'utente di selezionare due colonne numeriche dalla tabella.
4. Utilizzare i dati selezionati per creare un grafico a dispersione utilizzando Plotly.
5. Creare un file Dockerfile per la propria applicazione, specificando le dipendenze necessarie e i comandi per eseguire l'app.
6. Utilizzare il comando docker build per creare un'immagine Docker dell'app.
7. Utilizzare il comando docker run per eseguire l'immagine Docker e distribuire l'app su una piattaforma di cloud computing.
8. Configurare l'ambiente di destinazione in modo che sia in grado di accedere alla tabella auto.

# # Import necessary libraries

import dash

import dash\_core\_components as dcc

import dash\_html\_components as html

import pandas as pd

# # Connect to the database

df = pd.read\_sql('SELECT \* FROM auto', conn)

# # Create a Dash app

app = dash.Dash()

# # Create a dropdown component to select columns

column\_options = [{'label': i, 'value': i} for i in df.columns if df[i].dtype in ['int64', 'float64']]

column\_selector = dcc.Dropdown(id='column-selector', options=column\_options, value=df.columns[0])

# # Create a scatter plot using the selected columns

scatter\_plot = dcc.Graph(id='scatter-plot')

# # Define the layout of the app

app.layout = html.Div([

html.H1('Auto Database Scatter Plot'),

html.Div([

html.Label('Select X-axis Column'),

column\_selector,

]),

html.Div([

html.Label('Select Y-axis Column'),

column\_selector,

]),

scatter\_plot

])

# # Define a callback to update the scatter plot

@app.callback(

dash.dependencies.Output('scatter-plot', 'figure'),

[dash.dependencies.Input('column-selector', 'value')])

def update\_scatter\_plot(columns):

x\_column = columns[0]

y\_column = columns[1]

return {

'data': [{

'x': df[x\_column],

'y': df[y\_column],

'type': 'scatter'

}],

'layout': {

'title': f'{x\_column} vs {y\_column}'

}

}

# # Run the app

if name == 'main':

app.run\_server()

# # Use an official Python runtime as the base image

FROM python:3.8

# Set the working directory

WORKDIR /app

# Copy the requirements file into the container

COPY requirements.txt .

# Install the dependencies

RUN pip install -r requirements.txt

# Copy the application code into the container

COPY . .

# Run the app

CMD ["python", "app.py"]

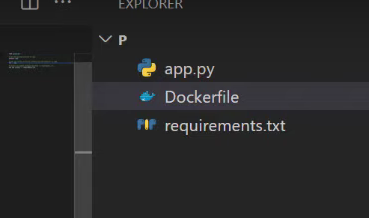
# SVILUPPO ESEMPIO

# SU VIRTUAL STUDIO CODE -> CREO IL FILE DOCKER

Dopo aver creato il file. App.py: Ho bisogno di un file txt requirment.txt e docker

-nel requirement…ci mette la versione del quale ha bisogno per dare l’app.. Immagine che contiene testo

Descrizione generata automaticamente



Scrive semplicemente dockerfile..

E ci scrive la versione di python

Immagine che contiene testo

Descrizione generata automaticamente

Immagine che contiene testo

Descrizione generata automaticamente

INSTALLARE SU VISUAL STUDIO L’ESTENSIONE DI DOCKER

LANCIA IL CONTEINER:

Immagine che contiene testo

Descrizione generata automaticamente

## FILE DI ESEMPIO COLLEGARE DOCKER A PYTHON

<https://github.com/phenixita/docker-python-101-basic>